This work has been supported by the CAM Project S2009/DPI-1559/ROBOCITY2030 II, developed by the research team RoboticsLab at the University Carlos III of Madrid.

Path Planning for Mobile Robot Navigation using Voronoi Diagram and Fast Marching

Santiago Garrido sgarrido@ing.uc3m.es
Departamento de sistemas y automática
Universidad Carlos III de Madrid
Leganés, 28914,Spain

Luis Moreno moreno@ing.uc3m.es
Departamento de sistemas y automática
Universidad Carlos III de Madrid
Leganés, 28914,Spain

Dolores Blanco dblanco@ing.uc3m.es
Departamento de sistemas y automática
Universidad Carlos III de Madrid
Leganés, 28914,Spain

Piotr Jurewicz pjurewic@ing.uc3m.es
Departamento de sistemas y automática
Universidad Carlos III de Madrid
Leganés, 28914,Spain

Abstract

For navigation in complex environments, a robot needs to reach a compromise between the need for having efficient and optimized trajectories and the need for reacting to unexpected events. This paper presents a new sensor-based Path Planner which results in a fast local or global motion planning able to incorporate the new obstacle information. In the first step the safest areas in the environment are extracted by means of a Voronoi Diagram. In the second step the Fast Marching Method is applied to the Voronoi extracted areas in order to obtain the path. The method combines map-based and sensor-based planning operations to provide a reliable motion plan, while it operates at the sensor frequency. The main characteristics are speed and reliability, since the map dimensions are reduced to an almost unidimensional map and this map represents the safest areas in the environment for moving the robot. In addition, the Voronoi Diagram can be calculated in open areas, and with all kind of shaped obstacles, which allows to apply the proposed planning method in complex environments where other methods of planning based on Voronoi do not work.

Keywords: Voronoi Diagram, Path Planning, Fast Marching.

1. INTRODUCTION

Since the 1980s mobile robot motion planning problems have become an important research topic that has attracted the attention of many researchers who have worked extensively to obtain efficient methods to solve these problems. This issue has been dealt with in two general ways: one approach has concentrated on solving motion planning problems using a previously known global environment or obstacle information and the robot characteristics, whilst the second approach has concentrated on planning motion using local sensor information and the robot characteristics.

In recent years, sensor-based path planning has emerged as a compromise between the need for reacting to unexpected environmental events and the need for having efficient optimized trajectories. Different variations of classical methods have been proposed to achieve an operative sensor-based path planning. One of these techniques is the Voronoi-based path planning. In order to use a sensor-based method an efficient and fast algorithm is required to extract and follow Voronoi Diagrams.

The Voronoi Diagram has been studied by many researchers. The advantages of Voronoi-based path planning are that it diminishes the dimension of the problem to one and that the trajectories follow the maximum clearance map. This means that the trajectories are the safest ones. Moreover, the Voronoi-based navigation reminds of topological navigation and, because of that, it is similar to the human one in some aspects. However, most Voronoi-based methods have the difficulty of calculating the Voronoi Diagram by studying lines and polygons, finding the vertices and nodes, and creating a tree to find the path. In addition, the method is not very efficient in three dimensions and it does not work at all in higher dimensions. As an alternative to solve these problems, we propose the use of image processing methods.

In order to calculate the trajectory, the new Motion Planning method is based on the combination of a Voronoi Diagram and the expansion of a wave from the start point to the goal following the Voronoi Diagram.

The proposed sensor-based planning method uses a combination of sensor information and the known map of the environment. The Voronoi Diagram is built using image methods (skeletonization) based on the Breu method in 2D [7] and Gagvani in 3D [16]. This Voronoi Diagram (skeleton) is dilated and inverted to obtain a viscosity map. The next step is to calculate the trajectory in the image generated by the thick Voronoi Diagram using a wave expansion. The wave expansion is calculated solving the Eikonal equation for which the Fast Marching Method has been used, though there are other similar ones, such as the Tsiksitlis [54] method, MCC [32], and others.

Then, the obtained path verifies the smoothness and safety considerations required for mobile robot path planning.

The approach implemented in this work is based on the potential field method of the wave expansion, which repels a robot away from obstacles and towards the goal using a carefully designed artificial potential function. The main advantages of this method are:

- It is easy to implement.
- The computation time is low, so this algorithm works in Real-Time.

2. RELATED WORK

2.1 Voronoi Diagram

The Voronoi concept has been used for four centuries. In his Traité de la Lumiére published in 1644, Descartes uses diagrams similar to Voronoi to show the disposition of matter in the solar system and its environment. Algorithms of Voronoi Diagrams have been appearing since the 1970s. See the surveys by Aurenhammer [2,3], de Berg [5], and Okabe [39] on various algorithms, applications, and generalizations of Voronoi Diagrams.

The Generalized Voronoi Diagram is the set of points where the distance to the two closest objects is the same [11]. Many algorithms have been proposed for constructing Generalized Voronoi Diagrams using the distance information provided by different external sensors like sonar sensors, laser scanners and stereo cameras. The method for construction of a Generalized Local Voronoi Diagram (GLVG) applied by Mahkovic in [31] uses measurements from a laser scanner. First, the points that belong to the same object are clustered, then the Voronoi Diagram is generated, and finally, the edges outside the visible region and those which both side generator points belonging to the same object are deleted. In [53], Sudha constructs a Voronoi Diagram from a binary image of the workspace obtained using a digital camera.

One class of algorithms involves incremental construction procedures based on the exploration of an unknown environment. It is the case of the approach proposed by Nagatani, Choset, and Thrun [35,9,11]. This algorithm does not consider the encoder errors and it cannot be used in a large environment.

There are some path planning algorithms inspired by the Voronoi Diagram concept, such as the MAPRM method [57], which retracts sampled configurations onto the medial axis (or Generalized Voronoi Diagram). Also, the EquiDistance Diagram (EDD) [22] is based on the Voronoi roadmap idea, that is a roadmap formed by connecting the local maxima of a clearance function defined using distance functions. The main shortcoming of these methods is that their roadmap is constructed offline and the environment information must be provided in advance.

Breu et al. [7] present a linear time (and hence asymptotically optimal) algorithm for computing the Euclidean distance transform of a two-dimensional binary image. The algorithm is based on the construction and regular sampling of the Voronoi Diagram whose sites consist of the unit (feature) pixels in the image. Their algorithm constructs the Voronoi Diagram where it intersects the horizontal lines passing through the image pixel centers. They also discuss the extensions to higher dimensional images and to other distance functions. The method proposed in this paper uses, in its first part, this algorithm.

2.2 Potential Field-based Motion Planning

From a theoretical point of view, the motion planning problem is well understood and formulated, and there is a set of classical solutions capable of computing a geometrical trajectory that avoids all known obstacles. Most of these general methods are not applicable if the environment is dynamic or there are no modelled obstacles. Hence, to avoid the problem of executing an unrealistic geometrical trajectory, in which the robot can collide with objects, obstacle avoidance algorithms have been developed to provide a robust way of coping with the problem.

One of the classical methods for dynamically solving the collision avoidance problem is the potential field approach developed by O. Khatib [23]. This approach is based on the creation of an artificial potential field in which the target is an attractive pole and the obstacles are repulsive surfaces. The robot follows the gradient of this potential toward its minimum. The derived force induces a collinear and proportional acceleration enabling easy dynamic and kinematic control actions. This technique can be used at a global or local level depending on the information used to generate the potentials. The major advantage of this method is its simplicity and its capability of being used dynamically because of the easy treatment of fixed and mobile obstacles. Its major disadvantage is the possible existence of local minima and the oscillations for certain configurations of obstacles. In spite of this problem, this technique has been used extensively in reactive architectures because of its natural ability to encapsulate specific robot behaviors.

In [33], Melchior at al. find an optimal trajectory using the Fast-Marching technique and compare it with the A* algorithm.According to Melchior, the solutions obtained with the Fast-Marching technique are the true approximations of the continuous and optimal trajectories. Thus the level line curvature and the gradient orientation are continuous; the robot can be oriented quasi-continuously. However, the minimum length trajectory obtained by the A* algorithm is not the discrete approximation of the optimal length continuous trajectory.

The Fast Marching Method has been used previously in Path Planning by Sethian [49,48], but using only an attractive potential. This method has some problems. The most important one that typically arises in mobile robotics is that optimal motion plans may bring robots too close to obstacles (including people), which is not safe. This problem has been dealt with by Latombe [26], and the resulting navigation function is called NF2. Melchior, Poty and Oustaloup [33,42] present a fractional potential to diminish the obstacle danger level and to improve the smoothness of the trajectories; Philippsen [41] introduces an interpolated Navigation Function, but with trajectories too close to obstacles and without smooth properties; and Petres et al. [40] introduce efficient path-planning algorithms for Underwater Vehicles that take advantage of underwater currents. They developed an algorithm called FM^*, which is a continuous version of the A^* algorithm based on Fast Marching. The Voronoi-based Method presented in [18] tries to follow a maximum clearance map.

2.3 Grid-based Planning Methods

In the navigation of mobile robots, many methods consist on a grid-based representation of the environment. This representation can have occupied or free cells (binary representation), or each cell can have an associated weight that represents the difficulty of traversing that area.

This grid is usually approximated by a graph, with the nodes situated in the center of each cell. Many algorithms have been developed to find a path in the graph. Dijkstra's algorithm computes the optimal path between a single source point to any other point in the graph. This algorithm is indeed efficient, but suffers from metrication errors [13], as shown in Fig. 1. A^* uses a heuristic to focus the search from a particular start location towards the goal, and thus produces a path from a single location to the goal very efficiently [37]. D^*, Incremental A^*, and D^* Lite are extensions of A^* that incrementally repair solution paths when changes occur in the underlying graph [52]. These incremental algorithms have been used extensively in robotics for mobile robot navigation in unknown or dynamic environments.

Grid or graph based methods, such as Dijkstra's or A^*, calculate a navigation function constrained to movement choices along graph edges that are neither optimal for execution nor an exact solution. The graph based algorithms consider the image as an oriented graph in which a pixel is a node, and the 4 (or 8) connections to the neighboring pixels are the vertices of the graph.

These methods can not converge to the solution of a continuous problem. Even for very fine grids, the paths restrict the agent's heading to increments of $\pi/2$ or $\pi/4$. This results in paths that are suboptimal in length and difficult to traverse in practice. The different metrics used lead to very different results.
Frequently, a smoothing step is performed after planning to alleviate this problem, which yields to unsatisfactory results because it solves only locally the discrete nature of the problem.

Various efforts have been made to overcome these shortcomings, including increasing the available headings or approximating continuous paths (see [36] and [14]), using suboptimal variants of the search algorithms to reduce computation time [28], and efficiently updating existing solutions when new information is received [52], [25].

The fast marching method uses the L2 metric, instead of the L1 (Manhattan) of Dijkstra's similar methods, and it makes an intrinsic interpolation that allows it to solve the continuous eikonal efficiently equation and to give the shortest path.

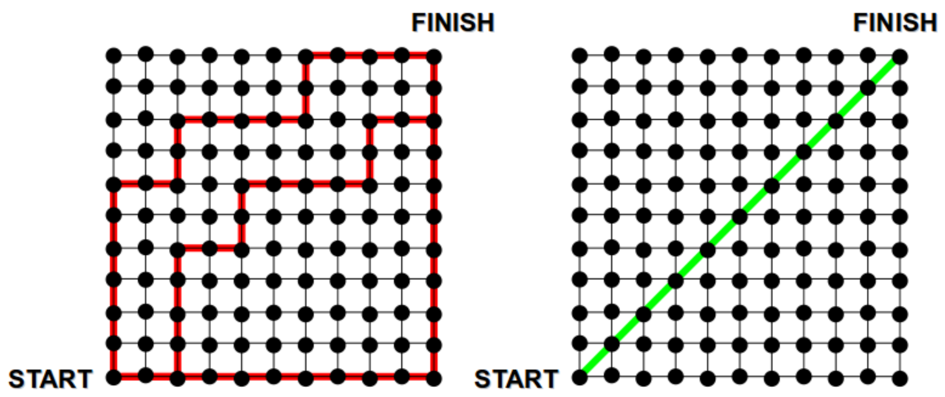

FIGURE 1: The Dijkstra Method gives multiple short paths but always following the conection between the graph points. Fast Marching Method gives the optimal diagonal path, because it interpolates, so the path obtained is the shortest.

3. INTRODUCTION TO THE VORONOI DIAGRAM AND SKELETON

The Voronoi Diagram concept is a simple but intuitively appealing one. Given a finite set of objects in a space, all locations in that space are associated with the closest member of the object set. The result is a partition of the space into a set of regions, Voronoi regions. The Generalized Voronoi Diagram is the frontier between these regions. Given its widespread use, it is not surprising that this concept has been discovered many times in many different places.

The close relationship between the Voronoi Diagram and the medial axis has already been pointed out in literature [38]. In the field of computer vision, the skeleton of an object or image is the Medial Axis Transform, which coincides with the definition of the GVD used in robotics that uses lines. This is not to be confused with the Voronoi Diagram of a discrete set of points: for each point x in the set P, there exists a boundary polygon enclosing all the intermediate points lying closer to x than to other points in the set $P,$ and the set of these polygons for a given point set is called the Voronoi Diagram.

A very illustrative definition of the skeleton is given by the prairie-fire analogy: the boundary of an object is set on fire and the skeleton is made up of the loci where the fire fronts meet and quench each other.

Based on Blum's definition of a skeleton [6], the skeleton S of a set D is the locus of the centers of maximal disks. A maximal disk of D is a closed disk contained in D that is interiorly tangent to the boundary δD and that is not contained in any other disk in D. Each maximal disc must be tangent to the boundary in at least two different points. With every skeleton point $s \in S$ we also store the radius $r(s)$ of its maximal disk. The skeleton $\sigma(D)$ is the set of centers of maximal disks in D. The skeleton is desired to be a 'graph-like' retraction of the original set. For a good overview of skeletonization techniques, see [51].

A class of skeletonization approaches is based on distance transform. In this paper algorithms based on distance transform (DT) are used for computing the skeleton (Voronoi Diagram) of the binary image that represents the environment. This associates every pixel in image space with its distance to the nearest point to it. The distance transform is important for natural neighbor interpolation of a function and for morphological operations on images, such as medial axis and skeleton computation, and polygon morphing. Many efficient algorithms exist for the computation of the distance transform, or a close approximation of it (meaning that the distances are not exactly the true distances) for images. The medial surface/axis can be defined as the locus of maximal circles (2D) or ball (3D). A particularly efficient algorithm that has been used for the calculation of the Voronoi Diagram (2D) in this work is due to Breu et al [7]. For its way of working it is not necessary to calculate vertices and nodes of the Voronoi Diagram and it can be used for all kinds of walls and obstacles, even curved ones. It runs in $O(m)$ time, where m is the number of image pixels. This algorithm is based on the construction and regular sampling of the Voronoi Diagram whose sites consist of the unit (feature) pixels in the image. The algorithm constructs the Voronoi Diagram where it intersects the horizontal lines passing through the image pixel centers. This algorithm has been used to compute the Voronoi Diagram in 2D environments.

An important drawback of the skeletonization methods is the existence of spurious *branches* on the generated skeletons. This undesired fact is caused by the irregularities of the boundary. Boundary noise would cause the Voronoi Diagram to be very dense which would lead to the need for appropriately pruned it to generate the medial axis. Ogniewicz [38] reduced skeletons formed from raster boundary points to a simple form. This was achieved by pruning the hair nodes of the skeleton until a given minimum circumcircle was reached. On the other hand, it has been demonstrated that hair nodes correspond to a location of minimum curvature on the boundary [4,6]. This means that for a sampled boundary curve, three adjacent points are co-circular, with their center at the skeleton hair. For the simplification of the skeleton, the hairs should be retracted to their parent node location. The central point of the three is moved towards the parent node until it coincides with the parent node circumcircle, resulting in smoothing outward-pointing salients of the boundary.

The process is repeated from the other side of the boundary, also retracting those salients. This may displace some of the points involved in the first smoothing step, but the process is convergent and a small number of iterations suffices to produce a smoothed curve having the same number of points as the original one, but with a simplified skeleton.

In order to extract the skeleton in 3D, the algorithm described in [16] has been implemented. That skeletonization method for 3D objects is based on a quasi-Euclidean distance transform. Once the distance transform value at every object point is known, a linded list of all the object points is created. One pass is done over the list to test if a local maximum of it satisfies the condition:

$$MNT_p < DT_p - TP \tag{1}$$

Where TP is a thinness parameter, DT_p is the distance transform of the voxel p, and MNT_p is the mean of the neighbors' distance transform. The effect of boundary noise is reduced by choosing a thinness parameter that removes most of the *'hairs'* in the skeleton.

4. INTUITIVE INTRODUCTION OF THE EIKONAL EQUATION AND THE FAST MARCHING PLANNING METHOD

The ideal planner always drives the robot in a smooth and safe way to the goal point. In nature there are phenomena with the same way of working: electromagnetic waves. If there is an antenna at the goal point that emits an electromagnetic wave, then the robot could drive himself to the destination following the waves to the source. The concept of the electromagnetic wave is especially interesting because the potential has all the good properties desired for the trajectory, such as smoothness (it is C^∞) and the absence of local minima.

Maxwell's laws govern the propagation of the electromagnetic waves and can be modelled by the wave equation:

$$\frac{\partial^2 \phi}{\partial t^2} = c^2 \nabla^2 \phi \tag{2}$$

A solution of the wave equation is called a wave. The moving boundary of a disturbance is called a wave front. The wave front can be described by the Eikonal equation. The reasoning that demonstrates it can be followed in [12].

In Geometrical Optics, Fermat's *least time principle* for light propagation in a medium with space varying refractive index $\eta(\boldsymbol{x})$ is equivalent to the Eikonal equation and can be written as $||\nabla\Phi(\boldsymbol{x})|| = \eta(\boldsymbol{x})$, where the Eikonal $\Phi(\boldsymbol{x})$ is a scalar function whose isolevel contours are normal to the light rays. This equation is also known as Fundamental Equation of the Geometrical Optics.

The Eikonal (from the Greek 'eikon', which means 'image') is the phase function in a situation for which the phase and amplitude are slowly varying functions of position. Constant values of the Eikonal represent surfaces of constant phase, or wave fronts. The normals to these surfaces are rays (the paths of energy flux). Thus the Eikonal equation provides a method for 'ray tracing' in a medium of slowly varying the refractive index (or the equivalent for other kinds of waves).
In the Sethian notation [47] the eikonal equation is written as

$$|\nabla T(x)|F(x) = 1 \tag{3}$$

where $T(\boldsymbol{x})$ represents the wave front (time), $F(\boldsymbol{x})$ is the slowness index of the medium, and $\boldsymbol{x} = (x, y)$ in 2D or $\boldsymbol{x} = (x, y, z)$ in 3D.

The theory and the numerical techniques known as Fast Marching Methods are derived from an exposition to describe the movement of interfaces, based on a resolution of the equations on partial differential equations as a boundary condition problem. They have the merit of unifying the ideas relative to the evolution of fronts in a general framework.

For the formulation of a front in evolution, having considered a generic limit, we introduce, for example, a unidimensional expansion, one curve in two dimensions (see Fig. 1), or one surface in three dimensions, which separates one region from another.

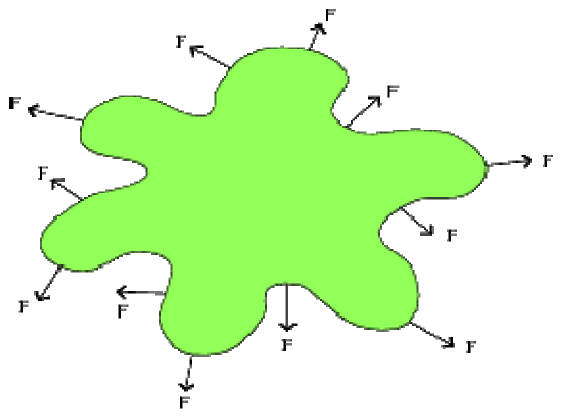

FIGURE 2: Wave front propagating with velocity F

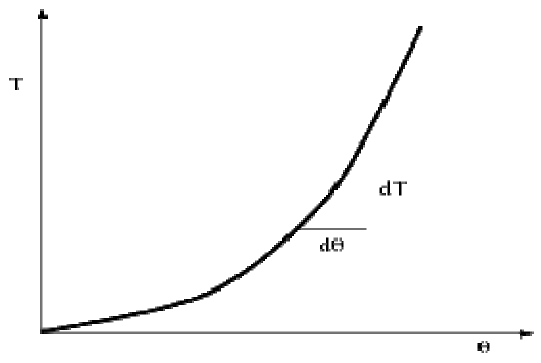

FIGURE 3: Formulation of the arrival function $T(\theta)$, for a unidimensional wavefront

Let us imagine that the curve or surface moves in its normal direction with a known speed F. The objective would be to follow the movement of the interface while this one evolves. A large part of the challenge in the problems modelled as fronts in evolution consists on defining a suitable speed that faithfully represents the physical system.

One way to characterize the position of a front in expansion is to compute the time of arrival T, in which the front reaches each point of the underlying mathematical space of the interface. It is evident that for one dimension (see fig. 2) we can obtain the equation for the arrival function T in an easy way, simply considering the fact that the distance θ is the product of the speed F and the time T.

$$\theta = F \cdot T$$

The spatial derivative of the solution function becomes the gradient

$$1 = F\frac{dT}{d\theta}$$

and therefore the magnitude of the gradient of the arrival function $T(\theta)$ is inversely proportional to the speed.

$$\frac{1}{F} = |\nabla T|$$

For multiple dimensions, the same concept is valid because the gradient is orthogonal to the level sets of the arrival function $T(\theta)$.

This way, we can characterize the movement of the front as the solution of a boundary condition problem. If speed F depends only on the position, then the equation $\frac{1}{F} = |\nabla T|$ can be reformulated as the eikonal equation

$$|\nabla T|F = 1. \qquad (4)$$

As a simple example we defined a circular front $\gamma_t = \{(x,y)/T(x,y) = t\}$ for two dimensions that advance with uniform speed. The evolution of the value of the arrival function $T(\theta)$ can be seen as time increases (i.e. $T = 0$, $T = 1$, $T = 2$, ...), and the arrival function comes to points of the plane in more external regions of the surface (see fig. 3). The boundary condition is that the value of the wave front is null in the initial curve.

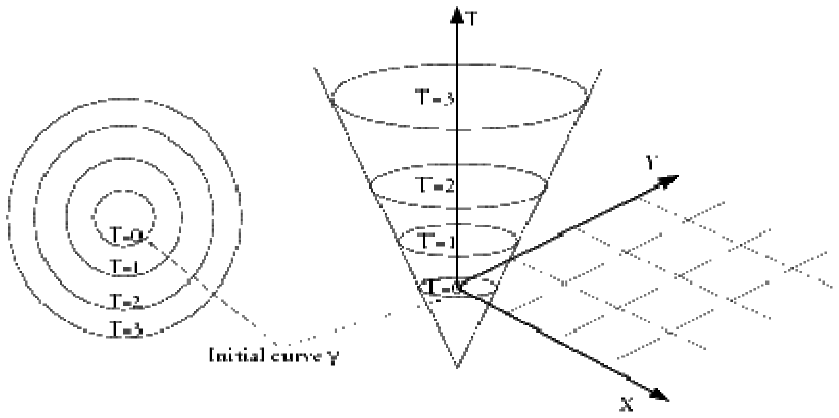

FIGURE 4: Movement of a circular wave front, as a problem of boundary conditions

The direct use of the Fast Marching Method does not guarantee a smooth and safe trajectory. From the way the front wave is propagated the shortest geometrical path is determined. This makes the trajectory unsafe because it touches the corners, walls, and obstacles, as shown in Fig. 4. This problem can be easily solved by enlarging the obstacles, but even in that case the trajectory tends to go close to the walls and it is not sufficiently smooth or safe enough.

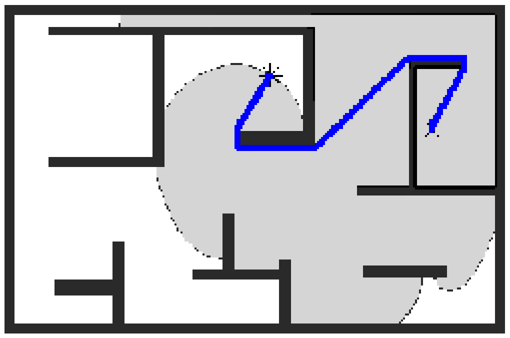

FIGURE 5: Trajectory calculated by Fast Marching Method directly. The trajectory is not safe because it touches the corners and walls.

The image 4 shows that the Fast Marching Method can not be used directly *Fast Marching Methods* are designed for problems in which the speed function never changes sign, so that the front is always moving forward or backward. This allows to convert the problem into a stationary formulation, because the front crosses each grid point only once. This conversion into a stationary formulation, in addition to a whole set of numerical tricks, gives it a tremendous speed to the method.

The distance given by the wave propagation of the Fast Marching Method is the Geodesic distance, which is different from the Euclidean Distance, as shown in Fig. 5. In that figure it is possible to observe how the points on the left foot are farther from the start when Geodesic distance is used. The Geodesic distance is the length of the shortest path between the two points.

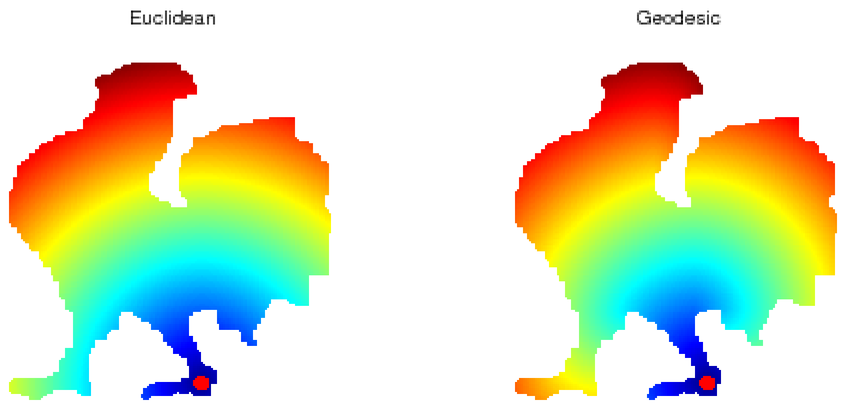

FIGURE 6: Difference between the Euclidean distance and the Geodesic distance calculated with the Fast Marching Method. The Geodesic distance is the distance of the minimum length inside the figure path and the Euclidean distance is the straight line distance

Since its introduction, the Fast Marching Method approach has been successfully applied to a wide set of problems that arise in geometry, mechanics, computer vision, and manufacturing processes, see Sethian [48] for details. Numerous advances have been made to the original technique, including the adaptive narrow band methodology [1] and the Fast Marching Method for solving the static eikonal equation ([49],[48]). For further details and summaries of level set and fast marching techniques for numerical purposes, see [48].

5. IMPLEMENTATION OF THE METHOD

This method operates in two main steps. The first step consists on the calculation of the Voronoi Diagram of the 2D or 3D map of the environment (that is, the cells located equidistant to the obstacles). This process is done by means of morphological operations on the image of the environment map. In the second step, the Fast Marching Method is used to create a potential T(x) that represents the propagation of an electromagnetic wave. This potential is used to calculate the trajectories based on the potential surface defined by the slowness map. To be more precise, the planner follows the following steps:

1. **Modelling.** A grid-based map of the environment is the model to be used. The a priori known map (if it is known) and the sensor information are integrated in an updated image of the environment with black and white cells. The only criteria to select the grid size is the computation time. The grid does not need to be uniform, being possible to use a triangular mesh.

2. **Object enlarging.** In order to avoid unrealistic trajectories, the objects and walls are enlarged by the radius of the mobile robot before the calculation of the Voronoi Diagram to ensure it neither collides with obstacles or walls nor accepts passages narrower than the robot size. This allows to consider the robot as a point from now on.

3. **Filter.** In order to remove the small *'hairs'* of skeletons, an averaging filter is applied. This filter is applied over the binary image to eliminate the corners and avoid the appearance of hairs.

4. **Voronoi Diagram.** Morphological image processing techniques are applied to the map to build the Voronoi Diagram. To be exact, a skeletonization of the image based in the Breu method (2D) [7] and Gagvani's (3D) [16] is applied in order to obtain the Voronoi Diagram as shown in Fig. 5. This skeleton is an image in which the points of the Voronoi diagram are black and the others white, but there are no information of vertices and nodes. It is possible to calculate the nodes, but it is more practical to search the path over the image with the Fast Marching method, because the calculation time is smaller and it works even with curved obstacles and walls.

5. **Dilatation** After that, a dilatation is done in order to have a thick Voronoi Diagram (see Fig. 5) wherein to calculate the propagation of the wave front of the Fast Marching Method. This is done in order to obtain two advantages. First, the sharp angles between segments of the diagram are smoothed, which improves the continuity of the generated trajectories and frees them from sharp angles. Second, the Voronoi Diagram is thickened in order to eliminate excessive narrowing, and thereby to allow a better propagation of the wave front when applying the Fast Marching Method. This way, the trajectory can be calculated.This thickening of all the lines of the Voronoi Diagram is done in the same amount, but the grade of dilatation is the only design parameter of the method. It can be small enough to have no isolated points or big enough to have a tube in which non-holonomic vehicles maneuver (see non-holonomic section) .

6. **Viscosity Map.** In order to apply the FMM, it is necessary to invert the image (see Fig. 1) to obtain a viscosity (or slowness) map since the wave travels faster in the clearer zones and slower in the darker ones.

7. **Fast Marching Method.** The next step is to calculate the trajectory in the image generated by the Voronoi Diagram using a wave expansion. The wave equation is approximated using the paraxial approximation used in optics. This way, the Eikonal equation is obtained. To solve the Eikonal equation the Fast Marching Method has been used. The Fast Marching Method is used to create a potential T(x) that represents the propagation of an electromagnetic wave in a viscosity map, to which the time is added as the third axis in 2D or the fourth axis in 3D. The origin of the wave is the goal point, which continues propagating until it reaches the current position of the robot. The Fast Marching Method produces a potential with a funnel-shaped surface, in which the time is the last axis, i.e. ,$T(x)$.

8. **Path Calculation.** This potential is used to calculate the trajectories based on the potential surface defined by the slowness map using the gradient method with the current position of the robot as the starting point and the goal position as the final point (see Fig. 1). The gradient determines the direction of travel after FMM has calculated $T(x)$.

The algorithm steps are summarized in the flowchart in Fig.5. The trajectory is obtained inside the safest areas provided by the thickened Voronoi Diagram and is properly smooth, especially at the angles, since the Fast Marching Method selects a continuous path in gradient terms. Moreover, the path extraction is very fast because the Fast Marching Method propagates in an almost unidimensional curve (although this is an approximation since the Voronoi Diagram is thickened in the perpendicular direction to the diagram curves).

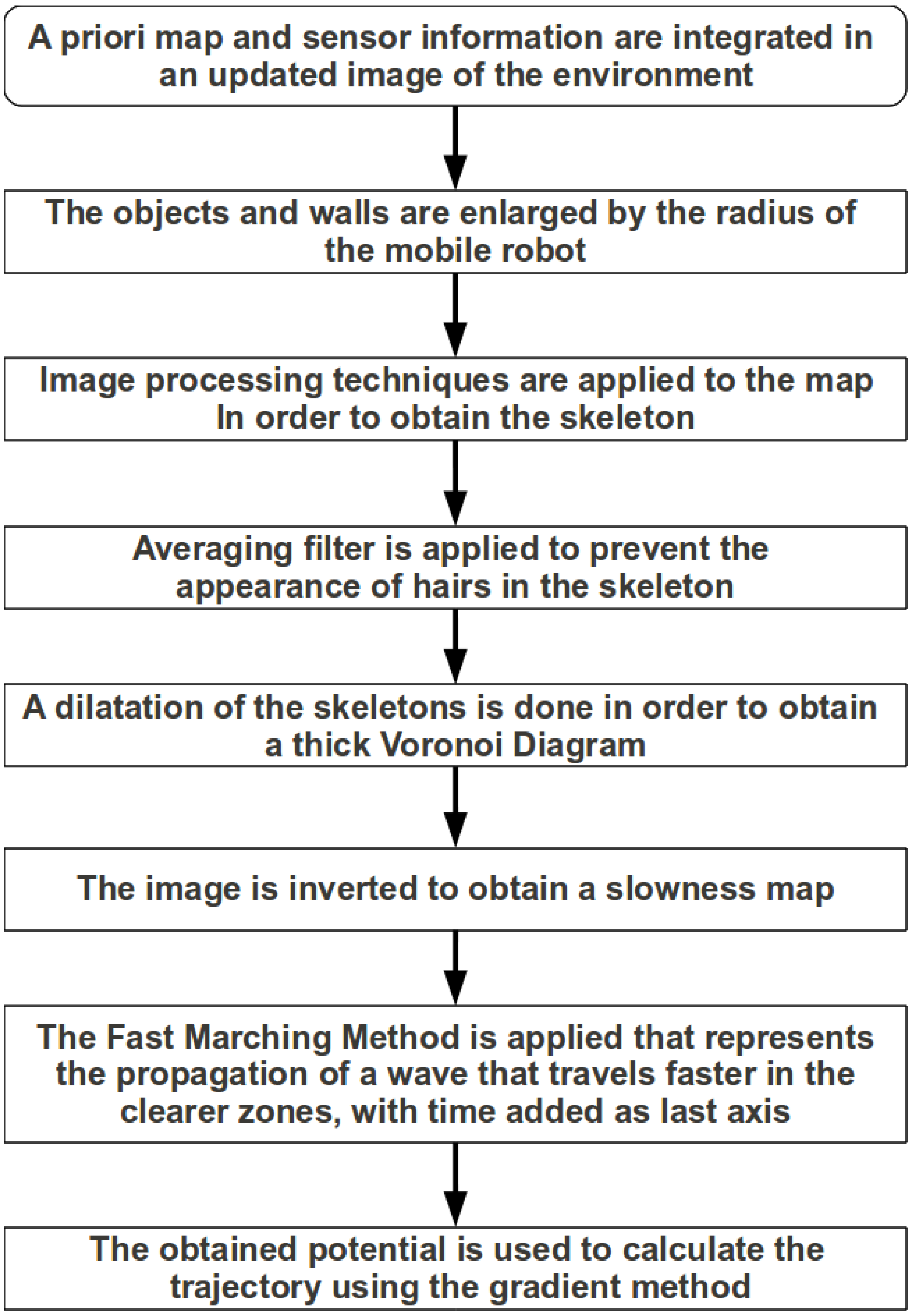
A priori map and sensor information are integrated in an updated image of the environment
The objects and walls are enlarged by the radius of the mobile robot
Image processing techniques are applied to the map In order to obtain the skeleton
Averaging filter is applied to prevent the appearance of hairs in the skeleton
A dilatation of the skeletons is done in order to obtain a thick Voronoi Diagram
The image is inverted to obtain a slowness map
The Fast Marching Method is applied that represents the propagation of a wave that travels faster in the clearer zones, with time added as last axis
The obtained potential is used to calculate the trajectory using the gradient method

FIGURE 7: Flowchart of the proposed planning algorithm

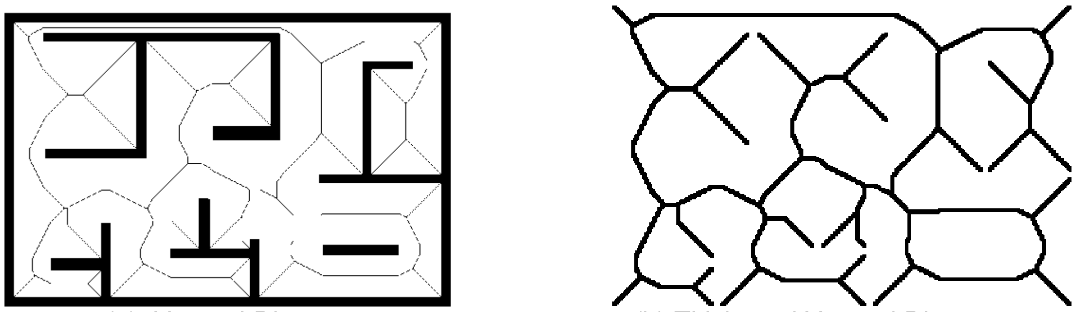
(a) Voronoi Diagram (b) Thickened Voronoi Diagram

FIGURE 8 : Map of the room used in the first experiment

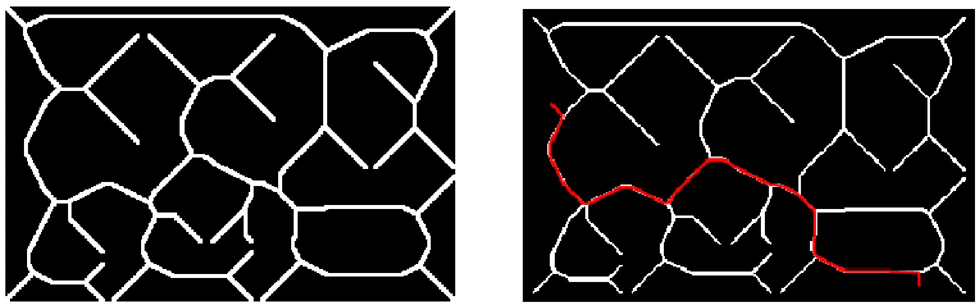

FIGURE 9: Trajectory calculated by Fast Marching Method in the inverted image of the thickened Voronoi Diagram of the room

The proposed method can also be used for sensor-based planning, working directly on a raw laser sensor image of the environment, as shown in Fig. 8 and 9. The data shown corresponds to the corner between two perpendicular aisles of the Carlos III University to the left of the robot, which is located in the center of the bottom of the scene. The image obtained from raw data may contain scattered pixels (see Fig. 7), and the dilatation of the image achieves the continuity and thickening of the lines representing the walls and obstacles (see Fig. 7). There are similar spurious lines behind the walls because the Voronoi diagram is made with an open image, but that path followed by the robot is correct. At this point the path planning steps described above are applied in the same sequence. The capability of using raw sensor data combined with the speed of the calculation allows this methodology to be used online to recalculate the trajectory at any time, avoiding this way dynamic obstacles and objects not present in the original map.

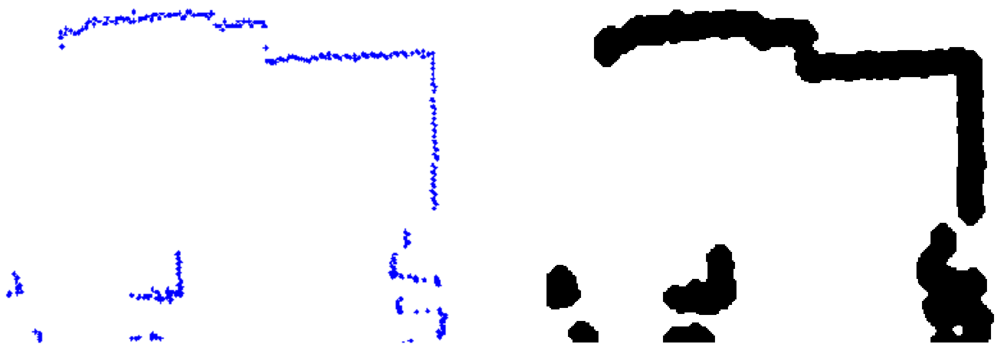

FIGURE 10: Data read by the robot (Local map)

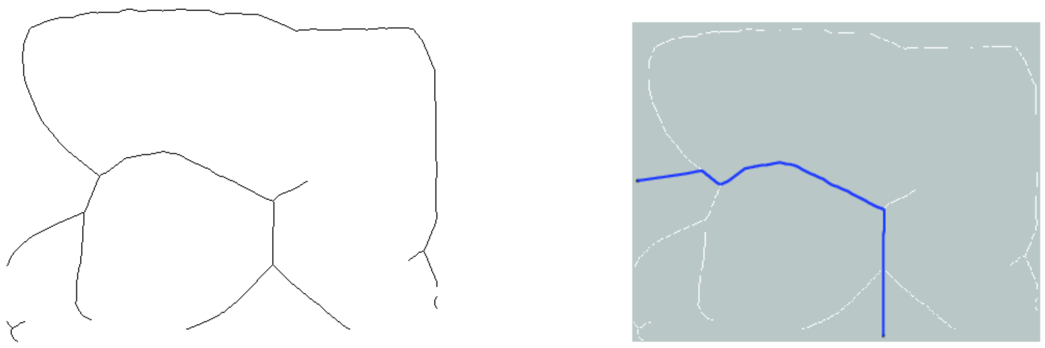

FIGURE 11: Voronoi Diagram and Trajectory of the Local map

6. RESULTS

The proposed method has been tested using the manipulator robot Manfred shown in Fig. 16. It has a coordinated control of all the degrees of freedom in the system (the mobile base has 2 DOF and the manipulator has 6 DOF) to achieve smooth movement. This mobile manipulator uses a sensorial system based on vision and 3D laser telemetry to perceive and model 3D environments. The mobile manipulator will include all the capabilities needed to localize and avoid obstacles and to navigate safely through the environment.

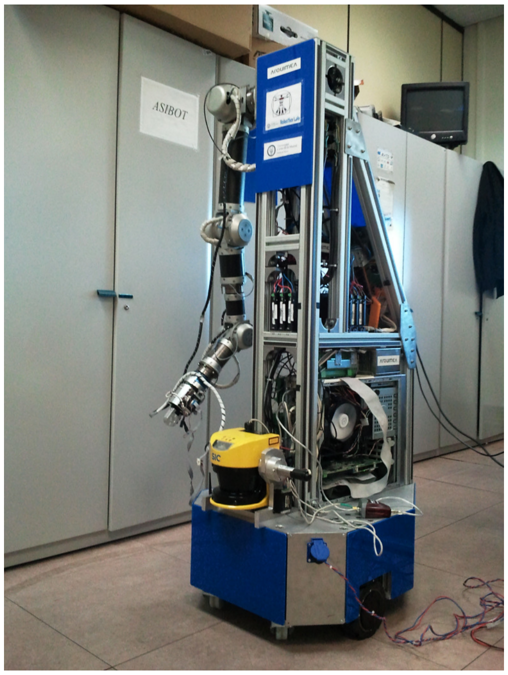

FIGURE 12: The robot Manfred has been used to test the algorithms

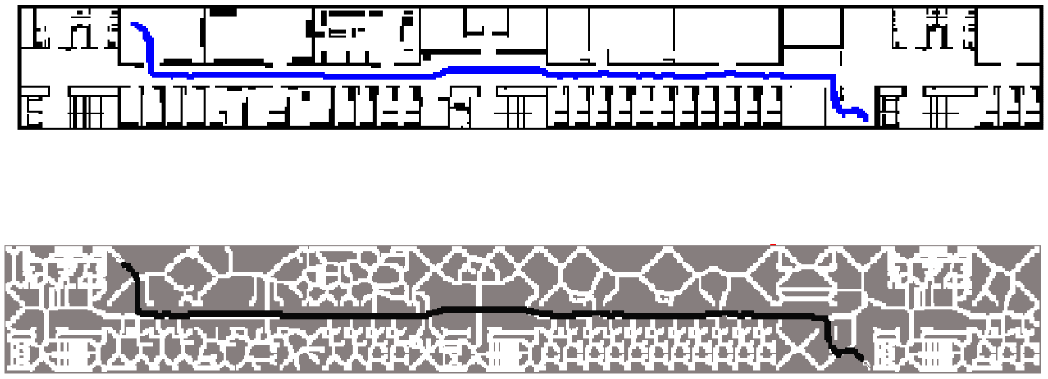

FIGURE 13: Trajectory calculated with Fast Marching over the Voronoi Diagram (Global map)

To illustrate the capabilities of the proposed method different tests are presented in this section. In the first test (room test with a resolution of 628×412 pixels), a difficult test room environment and the floor of the laboratory environment have been used (Fig. 6 and Fig. 7).

In the second test (laser test), the method is applied to a local environment path-planning task where the laser scanner measurement data are used to generate the trajectories, as it was explained in the previous section. Fig. 2 and 9 illustrate the achieved good trade-off between trajectory length, distances to obstacles and smooth changes in the trajectory. The Generalized Voronoi diagram of Fig. 13a goes behind the walls because the GVD is calculated on the image and in this image there is a possible path behind the walls.

The method has capabilities to generate adequate paths on a global scale as shown in Fig. 16. The results for the complete lab floor are shown in fig. 16 top (the environment map of the Robotics Lab of the Carlos III University) and fig. 16 bottom (the path obtained after applying the Fast Marching Method to the previous Voronoi Diagram image). The method provides smooth trajectories that can be used at low control levels without any additional smooth interpolation process.

The last test (lab test) is dedicated to illustrate the capability of the proposed method of adapting to a changing environment taking into account possible dynamic features of the environment such as moving obstacles and persons in the vicinity of the robot, as shown in Fig. 16. During the motion, the robot observes the environment with its laser scanner, introduces the new information in the map and plans a new trajectory. Local observations (obstacle in the middle of the corridor) generate modified trajectories in order to avoid the detected obstacles. In the bottom map of the figure the obstacles detected block the corridor and the sensor-based global planner generates a completely different safe trajectory. The dimensions of the environment are 116x14 m (the cell resolution is 12 cm). The image resolution is 966×120 pixels. For this environment (lab floor) the first step (the Voronoi extraction) takes 0.05 s in a Pentium 4 at 2.2 Ghz, and the second step (Fast Marching) takes 0.15 s for a long trajectory, which makes a total of 0.20 s to apply the Voronoi FM method, as shown in table 1.

The proposed method is highly efficient from a computational point of view because it operates directly over a 2D image map (without extracting adjacent maps), and due to the fact that Marching complexity is $O(n)$ and the Voronoi path calculation is also of complexity $O(n)$, where n is the number of cells in the environment map.

A great advantage of the method presented is that it can work with any kind of shaped objects, as demonstrated in Fig. 16. The curvature in the shape of the obstacle is not a restriction for the application of the method, and nor is the presence of concavities.

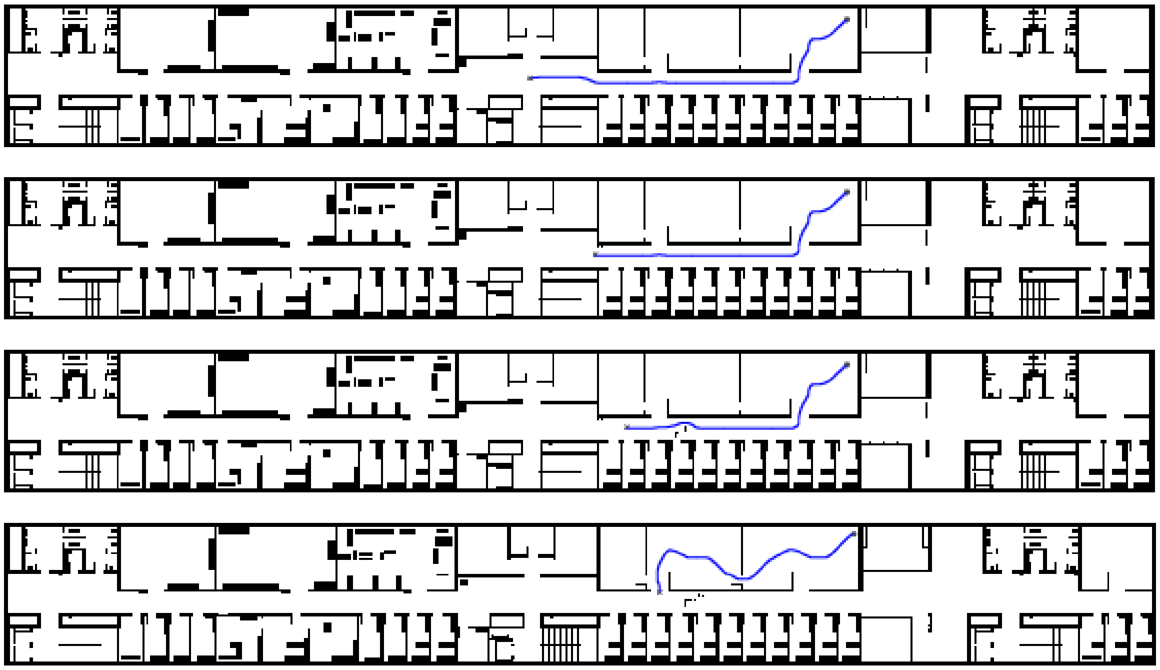

FIGURE 14: Evolution of the path when the robot reads information about the new obstacles that are not in the previous map and the robot can no pass throught the corridor

TABLE 1: Time comparison with other navigation Methods

Method	Laser(100x100 pixels)	Room(628x412 pixels)	Lab floor(966x120 pixels)
Voronoi FMM	0.017 s	0.446 s	0.20 s
Finite Differences	1.72 s	164.8 s	55.6 s
Finite Elements	1.34 s	33.6 s	16.2 s

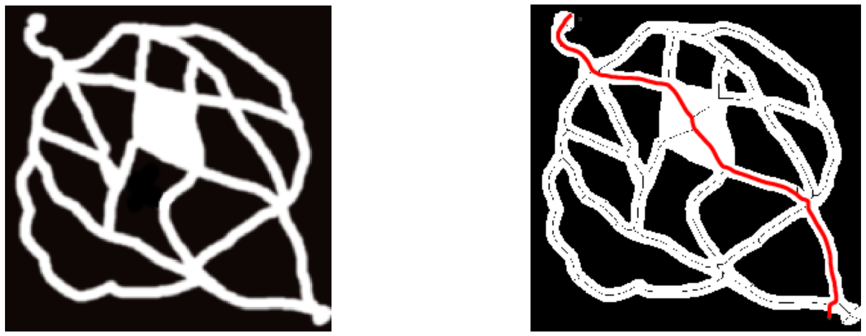

FIGURE 15: Voronoi Diagram and trajectory in an environment with curved shapes

In Fig. 16, the proposed method has been applied to a open environment, that is to say, to a environment map that is not limited totally by walls, in which there are circular obstacles. This map represents a corner with columns. As can be seen in the figure, the algorithm is perfectly able to make a correct planning of trajectories.

6.1 Comparison With Other Navigation Methods

One interesting method that calculates exact navigation functions is the Harmonic Functions method. Harmonic functions provide optimal potential maps for robot navigation in the sense of minimum length over the potential surface and they are the solutions of the Laplace equation $\nabla^2\Phi(x) = 0$. It has been implemented by using finite differences [43,44] and the finite elements method (FEM) [17] with a high level (usually 1) for the boundary conditions for walls and obstacles and a low level (usually 0) for objective. The finite differences method has been used to solve Laplace's equation with the successive over-relaxation (SOR) method, which is an iterative method and, consequently very slow. The finite elements method (FEM) is particularly useful when a robust approximation is sought to solve partial differential equations on non-homogeneous mesh. Solid mathematical foundations and a great deal of generality allow different implementations.

In table 1 a comparison between the different exact navigation methods is shown. This table compares the SOR method, the Finite Elements Method and the proposed Voronoi Fast

Marching method. As can be seen, the difference in time in comparison with the other methods is very significant.

6.2 Comparison With Other Navigation Method

Graph-based methods, such as Dijkstra, A^*, or D^*, can be considered to compute a navigation function constrained to movement choices along graph edges or discrete transitions between grid cells. They thus produce paths that are not optimal for execution. The Fast Marching method proposed gives a navigation function that measures distances in the continuous domain, due to its intrinsic interpolation (see Fig. 2).

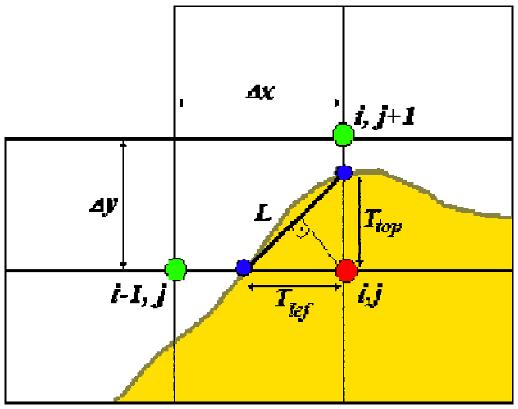

FIGURE 17: Calculation of Tij as distance of the point (i,j) to the line L (interpolation)

The graph methods are quite fast but they work in a graph. In our application the original map and the skeleton are binary images and they need to be transformed into its corresponding graph.
In table 2 a comparison with Dijkstra's and A^* methods is shown. As can be seen, these methods need a long time for the construction of the graph. The proposed method has also the advantage that it works in the continuous domain and its paths are smoother and not close to obstacles.

Table 2. Time comparison with Graph Methods

Method	Graph Generation time	Room(628x412 pixels)	Total time
Voronoi FMM	0.00 s	0.446 s	0.446 s
Dijkstra	2.82 s	0.845 s	3.665 s
A*	2.82 s	0.001 s	2.821 s

6.3 Extension to 3D

The proposed method works also in 3D cases. But not only that the Fast Marching Method can work also in higher dimensionality cases. The process is shown in Figure 3. The fist step is to extract the 3D skeleton, that could be considered as the Voronoi transform. That is done cropping the obstacles until the skeleton is obtained. Once that is done, the Fast Marching Method is launched. Fast Marching works for cases of 3,4 or even more dimensions. As in the two-dimensional case, the skeleton is interpreted as the boundary condition for the wave propagation. The process is shown in Figure 3, and as we can see the algorithms works well in the 3D-case.

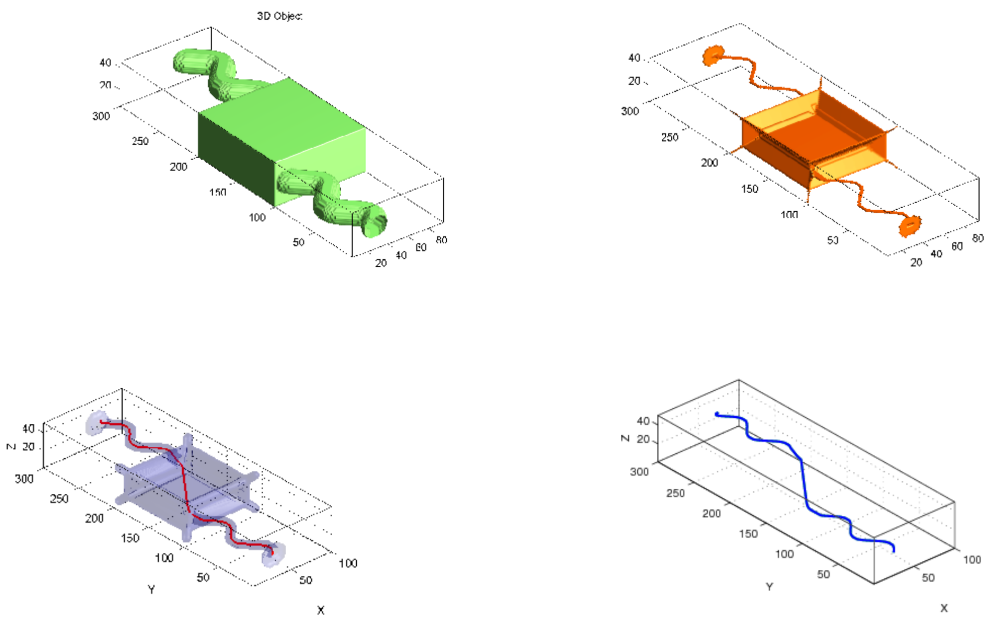

FIGURE 17: Trajectory calculated with the proposed method in 3D

7. CONCLUSION

A sensor-based path planner is presented in this paper. The proposed method is able to deal simultaneously with both global and local planning requirements. The advantages of the approach can be summarized by the fact that the trajectories obtained are smooth and safe, and at the same time, free of local traps due to the integration of the real-time sensor information in the recalculation of the path.

The method is easy to implement, the algorithm is very fast and can work online. It works in cluttered and changing environments with moving obstacles. The method is complete, i.e., the method is capable of finding a trajectory if it exists.

As demonstrated along this work, the method can perform in all types of environments without restrictions in the form of the obstacles. The planner works with curved forms, open environments (not totally enclosed by walls), and concavities.

The algorithm complexity is $O(n)$, where n is the number of cells in the environment map, which let us use the algorithm on line.

8. REFERENCES

[1] D. Adalsteinsson and J. Sethian, “A fast level set method for propagating interfaces,” *J. Comput. Phys.*, vol. 118, no. 2, pp. 269–277, (1995).

[2] F. Aurenhammer. “Voronoi Diagrams: A survey of a fundamental Geometric Data Structure,” *ACM Computing Surveys*, no. 23, 345-405, (1991).

[3] F. Aurenhammer and R. Klein, *Chapter 5 in Handbook of Computational Geometry*. Eds. J.R. Sack and J. Urrutia, pp. 201–290, (2000).

[4] H. Alt and O. Schwarzkopf, “The Voronoi Diagram of curved objects,” in *Proc. 11th Annual ACM Symposium on Computational Geometry*, pp. 89–97, (1995).

[5] M. de Berg, M. van Krefeld, M. Overmars and O. Schwarzkopf, *Computational Geometry: Algorithms ans Applications, 2nd rev.*. Springer, (2000).

[6] H. Blum, “A transformation for extracting new descriptors of shape,” in *Models for Perception of Speech and Visual Form*, W. W. Dunn, Ed.MIT Press, Cambridge, Mass., pp. 153–171, (1967).

[7] H. Breu, J. Gil, D. Kirkpatrick, and M. Werman, “Linear time euclidean distance transform algorithms,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 5, pp. 529–533, (1995).

[8] C. S. Chiang. “The Euclidean Distance Transform,” *Ph. D. Thesis, Dept. Comp. Sci., Purdue University*, (1992).

[9] H. Choset. “Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph,” *PhD thesis, California Institute of Technology, Pasadena, California*, March (1996).

[10] H. Choset and Burdic. “Sensor-Based Exploration: The Hierarchical Generalized Voronoi Graph,” *The International Journal of Robotics Research*, vol. 19, pp. 96–125, (2000).

[11] H. Choset et al., *Principles of Robot Motion: Theory, Algorithms, and Implementations*. The MIT Press, (2005).

[12] J. L. Davis, *Wave propagation in solids and fluids*. Springer, (1988).

[13] E. Dijkstra, “ A note on two problems in conexion with graphs,” *Numerische Mathematik*, vol. 1, pp. 269-271, (1959) .

[14] D. Ferguson and A. Stentz, *Advances in Telerobotics Field D*: An Interpolation-Based Path Planner and Replanner,* Springer Berlin, pp. 239-253, (2007).

[15] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robot. Automat. Mag.*, vol. 4, pp. 23–33, (1997).

[16] N. Gagvani and D. Silver, “Parameter controlled skeletonization of three dimensional objects,” *Technical Report CAIP-TR-216, Rutgers State University of New Jersey*, http://citeseer.ist.psu.edu/gagvani97parameter.html, (1997).

[17] S. Garrido, L.Moreno, “Robotic navigation using harmonic functions and finite elements,” in *Intelligent Autonomous Systems IAS’9*, pp. 94–103, (2006).

[18] S. Garrido, L. Moreno, D. Blanco, and F. Martin, “Voronoi Diagram and Fast Marching applied to Path Planning.” in *Proc of IEEE International conference on Robotics and Automation, ICRA’06*. pp.3049-3054. (2006).

[19] S. Garrido, L. Moreno and D. Blanco. “Sensor-based global planning for mobile robot navigation.” in *Robotica*. vol. 25. pp.189-199. (2007).

[20] S. Garrido, L. Moreno, and D. Blanco, *Exploration of a cluttered environment using voronoi transform and fast marching method*, Robotics and Autonomous Systems, doi:10.1016/j.robot.2008.02.003 (2008).

[21] M. Held. “Voronoi Diagrams and Offset Curves of Curvilinear Polygons.” Computer Aided Design, (1997).
[22] S.S. Keerthi, C.J. Ong, E. Huang, and E.G. Gilbert, “Equidistance diagram- a new roadmap method for path planning,” *In Proc. IEEE Int. Conf. On Robotics and Automation*, pp 682-687, (1999).

[23] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” *Int. J. Robot. Res.*, vol. 5, pp. 90–98, (1986).

[24] Y. Koren and J. Borenstein, “Potential field methods and their inherent limitations for mobile robot navigation,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1398–1404, (1991).

[25] S. Koenig and M. Likhachev, “ Improved fast replanning for robot navigation in unknown terrain,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, (2002).

[26] J.-C. Latombe, *Robot motion planning*. Dordrecht, Netherlands: Kluwer Academic Publishers, (1991).

[27] D.T. Lee, “Medial Axis Transform of a Planar Shape,” *IEEE Trans. Pattern Anal.Mach. Intell.,* PAMI-4, pp. 363-369, (1982).

[28] M. Likhachev, G. Gordon, and S. Thrun,“ Advances in Neural Information Processing Systems ARA*: Anytime A* with provable bounds on sub-optimality ,” *MIT Press*, (2003).

[29] S. R. Lindemann and S. M. LaValle,“ Simple and efficient algorithms for computing smooth, collision-free feedback laws”. International Journal of Robotics Research, (2007).

[30] S. R. Lindemann and S. M. LaValle, “Smooth feedback for car-like vehicles in polygonal environments”. In Proceedings IEEE International Conference on Robotics and Automation, (2007).

[31] R. Mahkovic and T. Slivnik “Generalized local Voronoi Diagram of visible region. *In Proc. IEEE Int. Conf. On Robotics and Automation*, pp. 349-355, Leuven, Belgium, May (1998).

[32] S. Mauch, “Efficient algorithms for solving static Hamilton-Jacobi equations,” Ph.D. dissertation, California Inst. of Technology, (2003).

[33] P. Melchior, B. Orsoni, O. Laviale, A. Poty, and A. Oustaloup, “Consideration of obstacle danger level in path planning using A* and fast marching optimisation: comparative study,” *Journal of Signal Processing*, vol. 83, no. 11, pp. 2387–2396, (2003).

[34] J. Minguez and L. Montano, “Nearness diagram navigation: collision avoidance in troublesome scenarios,” *IEEE Trans. Robot. and Automat.*, vol. 20, pp. 45–59, (2004).

[35] K. Nagatani, H. Choset, and S. Thrun. “Towards exact localization without explicit localization with the generalized voronoi graph,” *In Proc. IEEE Int. Conf. On Robotics and Automation*, pp. 342-348, Leuven, Belgium, May (1998).

[36] A. Nash, K. Danie, S. Koenig, and A. Felner, “Theta*: Any-Angle Path Planning,” *on Grids Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, p. 1177-1183, (2007).

[37] N. Nilsson, “Principles of artificial intelligence,” Palo alto, CA: Tioga Publishing Company. (1980).

[38] R. Ogniewicz and O. Kubler, “Hierarchic Voronoi Skeletons,” *Pattern Reconition*, (1995).
[39] A. Okabe, B. Boots, and K. Sugihara, “Spatial Tessellations: Concepts and Applications of Voronoi Diagrams”. John Wiley and Sons, Chichester, UK, (1992).

[40] C. Petres, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, D. Lane, “Path planning for autonomous underwater vehicles,” *IEEE Trans. on Robotics*, vol. 23, no. 2, pp. 331–341, (2007).

[41] R. Philippsen, “An interpolated dynamic navigation function,” in *2005 IEEE Int. Conf. on Robotics and Automation*, (2005).

[42] A. Oustaloup. A. Poty, and P. Melchior, “Dynamic path planning by fractional potential,” in *Second IEEE International Conference on Computational Cybernetics*, (2004).

[43] E. Prestes. Jr., P. Engel, M. Trevisan, and M. Idiart, “Exploration method using harmonic functions,” *Journal of Robotics and Autonomous Systems*, vol. 40, no. 1, pp. 25–42, (2002).

[44] E. Prestes. Jr., P. Engel, M. Trevisan, and M. Idiart, “Autonomous learning architecture for environmental mapping,” *Journal of Intelligent and Robotic Systems*, vol. 39, pp. 243–263, (2004).

[45] S. Quinlan and O. Khatib, “Elastic bands: connecting path planning and control,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 802–807, (1993).

[46] S. Quinlan and O. Khatib. “Efficient distance computation between non-convex objects,” in *IEEE Int. Conf Robot and Autom.*, pp. 3324-3329, (1994).

[47] J. A. Sethian, “A fast marching level set method for monotonically advancing fronts,” *Proc. Nat. Acad Sci. U.S.A.*, vol. 93, no. 4, pp. 1591–1595, (1996).

[48] J. A. Sethian, “Theory, algorithms, and applications of level set methods for propagating interfaces,” *Acta numerica*, pp. 309–395, (1996).

[49] J. Sethian, *Level set methods*. Cambridge University Press, (1996).

[50] R. Simmons, “The curvature-velocity method for local obstacle avoidance,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 3375–3382, (1996).

[51] R. W. Smith, “Computer processing of line images: A survey”, *Pattern Recognition*, vol. 20, no. 1, pp. 7-15, (1987).

[52] A. Stentz, “The focused D* Algorithm for Real-Time Replanning,” *Proceedings of the Internatinal Joint Conference on Artificial Intelligence (IJCAI)*, (1995).

[53] N. Sudha, S. Nandi, and K. Sridharan, “ A parallel algorithm to construct Voronoi Diagram and its VLSI architecture,” *In Proc. IEEE Int. Conf. On Robotics and Automation*, pages 1683-1688, (1999).

[54] J. N. Tsitsiklis, “Efficient Algorithms for Globally Optimal Trajectories,” *IEEE Transactions on Automatic Control*, vol. 40, pp. 1528-1538, (1995).

[55] I. Ulrich and J. Borenstein, “Vfh+: reliable obstacle avoidance for fast mobile robots,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 1572–1577, (1998).

[56] I. Ulrich and J. Borenstein, “Vfh*: local obstacle avoidance with look-ahead verification,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 2505–2511, (2000).

[57] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, “MAPRM: A probabilistic roadmap Planner with sampling on the medial axis of the free space”. *In Proc. IEEE Int. Conf. On Robotics and Automation*, pp. 1024-1031, (1999).

[58] L. Yang and S. M. LaValle, “A framework for planning feedback motion strategies based on a random neighborhood graph”. In *Proceedings IEEE International Conference on Robotics and Automation*, pp. 544–549, (2000).

[59] L.Yatziv, A. Bartesaghi, and G.Sapiro, “A fast O(n) implementation of the fast marching algorithm,” *Journal of Computational Physics*, vol. 212, pp. 393–399, (2005).